EXHIBIT 11

DOCKET NO: 0100157-00241

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

PATENT: 7,949,662

INVENTORS: DAVID A. FARBER
AND RONALD D. LACHMAN

FILED: DEC. 23, 2003 ISSUED: MAY 24, 2011

TITLE: DE-DUPLICATION OF
DATA IN A DATA PROCESSING
SYSTEM

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

PETITION FOR *INTER PARTES REVIEW* OF U.S. PATENT NO. 7,949,662 UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104

TABLE OF CONTENTS

I. MANDATORY NOTICES

A. Real Party-in-Interest

EMC Corporation ("Petitioner") is the real party-in-interest.

B. Related Matters

The '662 patent is one of an extensive patent family of continuation and divisional applications. Exhibit 1008 shows the patent family, with patents in red and blue including the '662 patent being asserted in the litigation *PersonalWeb Technologies LLC v. EMC Corporation and VMware, Inc.* (No. 6:11-cv-00660-LED) (E.D. Tex.), served on December 16, 2011.

Petitioner is also seeking *Inter Partes* Review of related U.S. Patents Nos. 5,978,791, 6,415,280, 7,945,539, 7,945,544, and 8,001,096, and requests that they be assigned to the same Board for administrative efficiency. Moreover, there are several continuing applications related to this family that remain pending (shown on Exhibit 1008 in green). Because they share a common disclosure with the '662 patent, these applications may be used as a basis to present patentably indistinct claims that may issue prior to the determination of the PTAB in this or related *Inter Partes* Reviews. The issuance of indistinct claims is at least inconsistent with Rule 37 C.F.R. 42.73(d)(3)(i) and would be an "end-around" the reasonable number of substitute claims that are permitted in an IPR proceeding. Petitioner respectfully

requests that the PTAB suspend from further prosecution, *sua sponte*, the applications in this related family, including the applications shown on Exhibit 1008 in green and any further applications that may be filed that depend from this family of patents. If the PTAB determines that that suspension should be requested by written motion, permission to file such a motion is requested at this time.

C. Counsel

Lead Counsel: Peter M. Dichiara (Registration No. 38,005)

Backup Counsel: David L. Cavanaugh (Registration No. 36,476)

Petitioners will request authorization to file a motion for Cynthia Vreeland to appear *pro hac vice*. Ms. Vreeland has more than 20 years litigation experience, and has worked with Petitioner EMC on IP litigation matters for more than 10 years. As such, Ms. Vreeland has experience and established familiarity with the technology at issue in the case. Petitioners intend to file a motion seeking admission of Ms. Vreeland to appear *pro hac vice* when authorized to do so.

D. Service Information

Email: Peter Dichiara, peter.dichiara@wilmerhale.com

Post and Hand Delivery: WilmerHale, 60 State St., Boston MA 02109

Telephone: 617-526-6466 Facsimile: 617-526-5000

E. Certification of Grounds for Standing

Petitioner certifies pursuant to Rule 42.104(a) that the patent for which review is sought is available for *inter partes* review and that Petitioner is not barred or estopped from requesting an *inter partes* review challenging the patent claims on the grounds identified in this Petition. Service of the complaint in *PersonalWeb Technologies LLC v. EMC Corporation and VMware, Inc.* (No. 6:11-cv-00660-LED) (E.D. Tex.) occurred on December 16, 2011. The one year time period for filing an *inter partes* review petition occurred on Sunday, December 16, 2012. This petition is timely filed the next business day, December 17, 2012. *See* 35 U.S.C. § 21.

II. OVERVIEW OF CHALLENGE AND RELIEF REQUESTED

A. Prior Art Patents and Printed Publications

Pursuant to Rules 42.22(a)(1) and 42.104 (b)(1)-(2), Petitioner challenges claim 30 of U.S. Patent No. 7,949,662 ("the '662 patent", Ex. 1001) as anticipated by or unpatentable in view of the following patents and printed publications:

1. S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," University of Tennessee

Technical Report CS-95-278 (Feb. 1995) ("Browne ", Ex. 1002).[1]

2. Kantor, "The Frederick W. Kantor Contents-Signature System Version 1.22," FWKCS122.REF (August 10, 1993) ("Kantor", Ex. 1004).[2]

---

[1]The Browne February 1995 publication qualifies as prior art under 35 U.S.C. § 102(a), and is used in this petition because it includes illustrations which facilitate explanation of the grounds of the invalidity. Petitioner also has attached as exhibits and included in its claim charts two earlier versions of this publication – S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," http://www.netlib.org/utk/papers/lifn/main.html (Nov. 11, 1994) (Exhibit 1006); and K. Moore et al., "An Architecture for Bulk File Distribution," Network Working Group Internet Draft (July 27, 1994) (Exhibit 1007). As Dr. Clark confirms in his declaration, the relevant disclosures are substantially the same. If the Patent Owner attempts to claim an earlier priority date of the challenged claims, Petitioner may rely on the earlier publications for invalidity, alone or in combination with the other references cited in this petition.

[2] Kantor's FWKCS user manual has been publicly and freely available continuously since August 1993. Kantor distributed the user manual with the FWKCS program as shareware and posted it online to electronic Bulletin Board Systems including "The Invention Factory" and "Channel 1" for an extended

3. M. Satyanarayanan et al., "Coda: A Highly Available File System for a Distributed Workstation Environment," IEEE Transactions on Computers, vol. 39, no. 4 (April 1990), pp. 447-459 ("Satyanarayanan II", Ex. 1026).[3]
4. Woodhill et al., U.S. Patent No. 5,649,196, entitled "System and Method For Distributed Storage Management on Networked Computer Systems Using Binary Object Identifiers," filed Nov. 9, 1995 as a continuation of application 85,596, filed July 1, 1993 ("Woodhill", Ex. 1005).
5. Ritchie and Thompson, "The UNIX Time-Sharing System," Comm. of the ACM, Vol. 17, No. 7 (July 1974) ("Ritchie", Ex. 1003).

B. There is a Reasonable Likelihood that Claim 30 of the '662 Patent is Unpatentable Under 35 U.S.C. §§ 102, 103

---

period of time, where it could be downloaded by anyone. As such the document was accessible to others in the relevant community of BBS users and system operators. (*See* Kantor at 3; *see also* 158-59; Ex. 1004.)

[3] This reference is referred to as "Satyanarayanan II," in order to distinguish it from another Satyanarayanan reference used in *Inter Partes* Reviews being filed by Petitioner on related patents.

Section VI below explains how the above-cited patents and printed publications create a reasonable likelihood that Petitioner will prevail with claim 30, the challenged claim. *See* 35 U.S.C. § 314(a). Indeed, that section together with the attached claim charts of Exhibits 1027, 1028, and 1029 and the Declaration of Dr. Douglas Clark, a Professor of Computer Science at Princeton University ("Clark Decl."; Ex. 1009), demonstrate that the challenged claim is anticipated by, or unpatentable in view of, each of these references.

C. Relief Requested

Petitioner requests cancellation of claim 30, the challenged claim, as unpatentable under 35 U.S.C. §§ 102 and 103.

III. Claim Construction

The claim terms should be given their "broadest reasonable construction in light of the specification." 37 C.F.R. § 42.100(b).

The claim terms can be understood by their plain and ordinary meanings except where construed in the specification. The specification includes the following constructions relevant to the challenged claims:

Claim Term	Construction
"data" and "data item"	"as used herein refer to sequences of bits. Thus a data item may be the contents of a file, a portion of a file, a page in

Claim Term	Construction
	memory, an object in an object-oriented program, a digital message, a digital scanned image, a part of a video or audio signal, or any other entity which can be represented by a sequence of bits" ('662 patent, col. 1, ll. 56-61, s*ee also* col. 1, l. 66-col. 2, l. 11 (indicating "data items" can include files, directories, records in the database, objects in an object-oriented programming, locations in memory or on a physical device or the like"); Ex. 1001.)
"file system"	"a collection of directories. A directory is a collection of named files – both data files and other directory files" ('662 patent, col. 5, ll. 35-37; Ex. 1001.)
"file"	"a named data item which is either a data file (which may be simple or compound) or a directory file. A simple file consists of a data segment. A compound file consists of a sequence of data segments. A data segment is a fixed sequence of bytes" ('662 patent, col. 5, ll. 37-42; Ex. 1001.)
"location"	"with respect to a data processing system, refers to any of a particular processor in the system, a memory of a particular processor, a storage device, a removable storage medium (such as a floppy disk or compact disk), or any other physical location in the system" ('662 patent, col. 5, ll. 53-

Claim Term	Construction
	58; Ex. 1001.)
"True Name, data identity, and data identifier"	"refer to the substantially unique data identifier for a particular item" ('662 patent, col. 5, ll. 61-65, see also col. 12, l. 39-col. 13, l. 37 (describing mechanism for calculating True Name using MD hash function); Ex. 1001.)

IV. OVERVIEW OF THE '662 PATENT

A. Brief Description

The '662 patent is directed to data storage systems that use "substantially unique data identifiers" – based on all the data in a data item and only the data in the data item – to identify and access data items. (*See, e.g.*, '662 patent, Title, Abstract, and col. 1, ll. 17-21; Ex. 1001.) The patent uses these identifiers to perform basic file management functions, such as requesting and obtaining computer files or other data items, and deleting unwanted duplicate copies of data items—admittedly old problems. (*See, e.g.*, '662 patent, Background of the Invention, col. 2, ll. 45-56; Ex. 1001.)

According to the patent, prior art systems identified data items based on their location or address within the data processing system. ('662 patent, col. 1, ll. 26-

33; Ex. 1001.) For example, files were often identified by their context or "pathname," that is, information specifying a path through the computer directories to the particular file (*e.g.*, C:\My Documents\Law School\1L\TortsOutline.txt). ('662 patent, col. 1, ll. 38-45; Ex. 1001.) The patent contends that all prior art systems operated in this manner: "In ***all*** of the prior data processing systems, the names or identifiers provided to identify data items. . . are ***always*** defined relative to a specific context," and "there is ***no*** direct relationship between the data names and the data item." ('662 patent, col. 2., ll. 12-13 (emphasis added); Ex. 1001.)

According to the patent, this prior art practice of identifying a data item by its context or pathname resulted in certain shortcomings. For example, with pathname identification, the same data name may refer to different data items, or conversely, two different data names may refer to the same data item. ('662 patent, col. 2, ll. 13-17; Ex. 1001.) Moreover, because there is no correlation between the contents of a data item and its pathname, there is no *a priori* way to confirm that the data item is in fact the one named by the pathname. ('662 patent, col. 2, ll. 18-21; Ex. 1001.) Furthermore, context or pathname identification may

more easily result in the creation of unwanted duplicate data items, e.g., multiple copies of a file on a file server.[4] ('662 patent, col. 1, ll. 44-55; Ex. 1001.)

The '662 patent purports to address these shortcomings. ('662 patent, col. 3, ll. 4-19; Ex. 1001.) It suggests that "it is therefore desirable to have a mechanism . . . to determine a common and substantially unique identifier for a data item, using only the data in the data item and not relying on any sort of context." ('662 patent, col. 3., ll. 4-9; Ex. 1001.) Moreover, "[i]t is further desirable to have a mechanism for reducing multiple copies of data items… and to have a mechanism which enable the identification of identical data items so as to reduce multiple copies." ('662 patent, col. 3, ll. 10-13; Ex. 1001.)

To do so, the '662 patent provides data identifiers that "depend[] on all of the data in the data item and only on the data in the data item." ('662 patent, col. 3, ll. 27-31; see also Field of the Invention ("This invention relates to data processing systems… wherein data items are identified by substantially unique identifiers which depend on all of the data in the data items and only on the data in the data

---

[4] For example, Alice and Bob both download the same copy of the James Bond movie *Goldfinger*. Alice saves her copy at "C:\Movies\Bond\Goldfinger.mov", and Bob saves his copy at "C:\Videos\007\Bond-Goldfinger.mov".

items"), col. 1, ll. 17-21; Ex. 1001.) The preferred embodiments use either of the well-known MD5 or SHA message digest hash functions[5] to calculate a substantially unique identifier from the contents of the data item. ('662 patent, col. 12, l. 39 – col. 14, l. 22; Ex. 1001.) The system first computes the 16-byte (128-bit) message digest of the data item and then appends the size of the data item to produce a 160-bit identifier. ('662 patent, Fig. 10A and col. 13, ll. 51-62; Ex. 1001.) The patent calls these context- or location-independent, content-based identifiers a "True Name" – a phrase admittedly "coined by the inventors." (U.S. Patent No. 6,415,280 Prosecution History, Response (Aug. 22, 2001), at 22; Ex. 1019.)

If a data item is large, it may include multiple components or "segments," representing the larger "compound data item." ('662 patent, col. 13, l. 63-col. 14, l. 6; Ex. 1001.) A True Name identifier is then computed for each of the segments based on a hash of the contents of the segment. ('662 patent, col. 14, ll. 6-14; Ex.

[5] A message digest or hash function transforms of a piece of data into a much shorter form by performing mathematical operations on its content. (*See, e.g.*, D. Banisar et al., The Third CSPR Cryptography and Privacy Conference at 509 (1993); Ex. 1010) The '662 patent admits that message digest functions were known. ('662 patent, col. 12, l. 66-col. 13, l. 3; Ex. 1001.)

1001.) A True Name identifier can then computed for the compound data item as a whole based on a hash of the segment hashes. ('662 patent, col. 14, ll. 6-14; Ex. 1001.)

With these identifiers, the patent asserts, "data items can be accessed by reference to their identities (True Names) independent of their present location." ('662 patent, col. 33, ll. 44-46; *see also* col. 33, ll. 65-67; Ex. 1001.) The actual data item corresponding to these location-independent identifiers may reside anywhere, e.g., locally, remotely, offline. ('662 patent, col. 33, ll. 46-54; Ex. 1001.) "Thus, the identity of a data item is independent of its name, origin, location, address, or other information not derivable directly from the data, and depends only on the data itself." ('662 patent, col. 3, ll. 30-33; Ex. 1001.)

In the preferred embodiments, the substantially unique identifiers are used to "augment" standard file management functions of an existing operating system. ('662 patent, col. 6, l. 66-col. 7, l. 6; Ex. 1001.) For example, a local directory extensions (LDE) table[6] is indexed by a pathname or contextual name of a file and

[6] According to the patent, a LDE table is a data structure which provides information about files and directories in the system and includes information in addition to that provided by the native file system. ('662 patent, col. 7, l. 66-col. 8,

also includes True Names for most files. ('662 patent, col. 7, l. 66-col. 8, l. 6; Ex. 1001.) A True File registry (TFR) lists True Names, and stores "location, dependency, and migration information about True Files." ('662 patent, col. 8, ll. 8-10, 13-15; Ex. 1001.) True Files are identified in the True File registry by their True Names, and can be looked up in the registry by their True Names. ('662 patent, col. 8, ll. 10-12; col. 23, ll. 42-43; Ex. 1001.) This look-up provides, for each True Name, a list of the locations, such as file servers, where the corresponding file is stored. ('662 patent, col. 33, ll. 52-54; *see also* col. 15, ll. 60-62; Ex. 1001.)

Data items may be shared by multiple entities, meaning each entity may have a different pathname for the data item as reflected in the LDE, but each entry also corresponds to the same contents identified by a True Name. Thus, deleting a data item deletes a reference to it by a particular entity (i.e., that entity no longer needs it, even though others still may). In operation, entries in an audit file reflect delete operations to be performed. ('662 patent, col. 16, 1. 27-col. 17, l. 36; Ex. 1001.). When a delete entry is processed, the "use count" for the True Name for that data item is reduced. ('662 patent, col. 17, ll. 12-36; Ex. 1001.). If the "use

---

l. 6; Ex. 1001.)

count" for that data item were one (so that after the reduction it will be zero), that means no other entities are using the data item and the underlying storage (*e.g.*, on disk) may be freed to store other data going forward. ('662 patent, col. 20, ll. 7-9; see also *id*. at col. 18, ll. 51-55; Ex. 1001.). If on the other hand, the user count were greater than one, that means even though the present entity is deleting its reference to the data item, other entities are still using it – in this case, the underlying storage is not freed because other entities are still using it. ('662 patent, col. 20, ll. 10-12; Ex. 1001.). An entry is made in an audit file to reflect the deletion. ('662 patent, col. 20, ll. 13-15; Ex. 1001.)

B. The Prosecution History of the '662 Patent

The '662 patent application was filed on December 23, 2003, based on an application that was originally filed on April 11, 1995. Roughly seven years after filing (and many office actions), the applicants submitted an amendment that included over 20 new claims, with no comment about the substance of those claims. (Amend. of Jan. 13, 2011 at 8-22; Ex. 1024.)

The application was subsequently allowed, and claim 109 was re-numbered to challenged claim 30. (Notice of Allowance; Ex. 1025.)

V. THE CHALLENGED CLAIM IS UNPATENTABLE

A. There is Nothing New About Deleting Files Using Content-based Identifiers

The '662 claims focus on the concept of using content-based identifiers to delete duplicate data items. Claim 30 of the '662 patent is reproduced below:

> 30. A computer-implemented deletion method operable in a file system comprising (i) a plurality of servers; (ii) a list indicating, for each of a plurality of files in the file system, a corresponding status,
>
> wherein, for each of a plurality of data items in the file system, said data items each consisting of a corresponding sequence of one or more parts; and
>
> wherein each data item has a corresponding digital data item identifier, said digital data item identifier for the data item being based, at least in part, on the contents of the data item, wherein two identical data items in the file system have the same digital data item identifier; and
>
> wherein each part is replicated on multiple servers of said plurality of servers; and
>
> wherein said list includes digital data item identifiers for data items for which changes are to be made in the file system,
>
> the method comprising the steps of:

> (A) obtaining a particular digital data item identifier of a particular data item, said particular digital data item identifier of said particular data item being obtained in response to an attempt to delete said particular data item in said file system;
>
> (B) updating a record in said list to reflect deletion of said particular data item from the file system, said record including the particular digital data item identifier to the list.

('662 patent, col. 43, ll. 28-36; Ex. 1001.) The basic limitations of the claim are straightforward. A data items is identified with a "data item identifier" based, at least in part, on its contents. This identifier is obtained in response to an attempt to delete the data item (step (A) of the claim). A record, including the data item identifier, is then updated to reflect deletion of the data item (step (B).) The preamble indicates that the system is a "file system," that the files are replicated on multiple servers, and that a list identifies data items for which changes are to be made by their content-based identifiers.

The applicants stated that they were entitled to these broad claims because "[i]n ***all*** of the prior data processing systems, the names or identifiers provided to identify data items . . . are ***always*** defined relative to a specific context," and "there

is ***no direct relationship*** between the data names and the data item." ('662 patent, col. 1, l. 66-col. 2, l. 4, col. 2, ll. 12-13, emphasis added; Ex. 1001.)

These representations were simply wrong. Prior data processing systems ***did use*** identifiers based on the contents of a data item or its segments – and not the context or pathname. In fact, these techniques were old and widely used. This is not surprising. The concept of using a mathematical function to create a "fingerprint" or "signature" for a data item based on the content of the data item predates the '662 patent by decades. For example, IBM developed one of the first hash tables in the 1950s (*see*, *e.g.*, G. D. Knott, Hashing functions, The Computer Journal 18 (1975), no. 3, at 274 (discussing "history of hashing"); Ex. 1011), and Professor Ron Rivest of MIT introduced the MD5 hash algorithm referenced in the '662 patent in the early 1990s. (*See, e.g.*, R. Rivest, "The MD5 Message-Digest Algorithm," Internet RFC 1321 (Apr. 1992); Ex. 1012.)

Moreover, Professor Ralph Merkle and others were partitioning data items into segments, calculating identifiers for the segments using a hash function, and then "hashing the hashes" to create top-level signatures (*i.e.*, identifiers for the data

items as a whole) by the 1970s.[7] (See Merkle, U.S. Patent No 4,309,569, entitled "Method of Providing Digital Signatures," filed Sept. 5, 1979, at col. 2, ll. 54-67 and Figure 1 (describes calculating signatures for a "vector of data items" by calculating signatures for segmented portions of the vector using a hash function, then combining the signatures using the same hash function) ("Merkle"); Ex. 1030.) "Hash trees" – also known as "Merkle trees" – were well known in the field long before the '662 patent.

These hashing functions take as input the data contained in a file, segment, or other data item, and produce a much smaller-sized output value, commonly called a "hash," "hash value," "message digest" ("MD"), or "checksum." (*See, e.g.*, McGraw-Hill Dictionary of Scientific and Technical Terms, (4th ed., 1989), at

---

[7] The idea of partitioning data into smaller segments (e.g., "pages" or "blocks") has been known for decades. (*See, e.g.*, B. Lampson and R. Sproull, "An Open Operation System for a Single-User Machine," ACM Operating System Review (Dec. 1979) at 100 ("The system organizes long-term storage (on disk) into files, each of which is a sequence of fixed-size pages" and "[t]he data bytes of the file are contained in pages 1 through n."); Ex. 1031; *see also* A. Tanenbaum, "Operating Systems Design and Implementation," Prentice-Hall (1987) at 256; Ex. 1032.)

860; Ex. 1013; *see also* B. Kaliski, "A Survey of Encryption Standards," IEEE Micro (Dec. 1993), pp. 74–81, at 77; Ex. 1014.) For example, a file that is a million bytes (or even much larger) in size can be used as input to produce a hash value that is a mere 16 bytes in length. Because of the mathematical properties of the function, the odds that two different files will produce the same 16 byte hash are extremely small: for example, with a 16 byte hash output, the odds that two randomly picked inputs have the same hash are 2^{-64}, or approximately one in sixteen billion billions. (B. Kaliski at 77; Ex. 1014.) Consequently, hashes are known as "signatures" or "fingerprints" because they identify data items with high reliability, just like signatures or fingerprints identify people with a high degree of certainty. (*See* D.R. McGregor and J.A. Mariani, 'Fingerprinting' – A Technique for File Identification and Maintenance, 12 Software Practice & Experience 1165 (1982), ("fingerprinting" technique "produce[s] a quasi-unique identifier for a file, derived from that file's contents. . .[t]he idea is to provide an identifying feature for every file, which is intrinsically distinctive, and analogous (hopefully) to a human's fingerprint."); Ex. 1017.)[8]

[8] This reference was central to the rejection of EP counterpart application EP0826181A1 with claims having a central feature of content-based identifiers.

The concept of deleting files is even older than the concept of hash-based identifiers – indeed, it is as old as the idea of files themselves. In fact, the particular approach discussed in the '662 patent was described in a seminal paper by Ritchie more than two decades earlier. (Ritchie and Thompson, "The UNIX Time-Sharing System," Comm. of the ACM, Vol. 17, No. 7 (July 1974) at 368-9 ("Ritchie") (Ex. 1003) (discussing Unix's use of an "i-list" comprised of "i-node" entries, each of which includes a "use count;" deleting "a file is done by decrementing the [use] count of the i-node. . . and erasing the directory entry. If the [use] count drops to 0, any disk blocks in the file are freed and the i-node is deallocated [thereby updating the i-list]."). Although the applicants suggested in their patent application that they were the first to utilize hash functions to identify

---

(Annex to the communication dated May 8, 2009; Ex. 1020.) The applicants amended the claims to emphasize a "licensing" limitation not found in the challenged claims (Reply to communication from the Examining Division dated November 18, 2009 at 4; Ex. 1021.), but this too was found unpersuasive and the rejection was maintained by the EPO. (Annex to the communication dated March 14, 2012 at 4; Ex. 1022) Following this rejection, Applicants withdrew the application from consideration. (Closing of Application dated June 14, 2012; Ex. 1023.)

data items for file management applications, including deleting files, others working in the field used them for the same purposes more than a decade before the '662 patent. For example, at least sixteen years before the '662 patent was filed, researchers were already using hash functions to determine whether two records were identical, and to delete duplicate records. (*See*, *e.g.*, Babb, Implementing a Relational Database by Means of Specialized Hardware, ACM Transactions on Database Systems, Vol. 4, No.1, at 2-4, March 1979; Ex. 1033; Bitton and DeWitt at 256 (commenting on the work of Babb); Ex. 1034.) File "fingerprinting" has long been known as a technique to identify files, and to check for duplicates. (*See* Rabin, Fingerprinting by Random Polynomials, Center for Research in Computing Technology, Harvard University, Report TR-15-81 at 1 and 9, 1981; Ex. 1015; *see also* Manber, at 3 (commenting on work of Rabin); Ex. 1016; McGregor and J.A. Mariani, at 1165 ; Ex. 1017.)

Many other printed publications and patents disclose and use identifiers exactly like those described and claimed in the '662 patent delete for exactly the same purposes. These publications disclose identifiers that are location- and context-independent, that are determined using only the contents of a data item,

and that are formed using identical algorithms to those mentioned in the '662 patent.

Browne: For example, researchers at the University of Tennessee and Bell Laboratories disclosed a system that created "location-independent file names" (or "LIFNs") to identify files on the Internet. (Browne at 3; Ex. 1002). LIFNs – like the identifiers in the '662 patent – uniquely identified files based on their contents, not their locations. (Browne at 3; Ex. 1002; *compare* '662 patent, col. 33, ll. 44-46 (True Names used to identify files "independent of their present location"); Ex. 1001.) LIFN <signatures> were computed as "the ascii form of the MD5 signature of the file" – the same function identified in the '662 patent. (Browne at 6; Ex. 1002; *compare* '662 patent, col. 12, ll. 66-67 (using MD5 or SHA); Ex. 1001.)

Browne specifically discussed both the mirroring and deletion of files. (Browne at 4-5; Ex. 1002.) A "LIFN database" maintained a list of list of LIFN <signatures> and corresponding file servers. (Browne at 6; Ex. 1002.) This database "accepts updates from file servers containing new locations for LIFNs, as well as requests to delete old LIFN-to-location mappings." (Browne at 5; Ex. 1002.) A file server could mirror a file by acquiring a copy of it and posting an update to a LIFN server. (Browne at 5; Ex. 1002.) Similarly, "[i]f a file server

moves or deletes a file, then it would post that information as well." (Browne at 5; Ex. 1002.)

Kantor: Dr. Frederick W. Kantor, a physicist from Columbia University, developed yet another example of context- and location-independent identifiers for the same purposes as the '662 patent. Dr. Kantor described a product called FWKCS that created "contents-signatures" for files based on their content. (Kantor at Preface 2; Ex. 1004.)[9] FWKCS used these contents-signatures to uniquely identify files on a bulletin board system ("BBS"), an online file system considered a precursor to the World Wide Web. (*Id.* at Preface 2; Ex. 1004) The contents-signature, as the name suggests, was based on a hash of the data contained in a file, just like the identifiers in the '662 patent. (*Id.* at 8; Ex. 1004; *compare* '662 patent, col. 15, ll. 37-48 and Figure 10A; Ex. 1001).

Kantor, like Browne, also specifically addressed the issue of compound data items, in particular, "zipfiles" containing a set of other files meant to be kept

[9] The three-page Preface section of Kantor's FWKCS user manual does not have individual page numbers. Citations to the Preface are labeled "Preface" to denote pages 1-3 of the Preface section. Otherwise, citations refer to the page numbers in the top-right margin of the remainder of the user manual.

together. (Kantor at Preface 1; Ex. 1004.) FWKCS created "zipfile contents-signatures" for these zipfiles, based on a hash of the contents-signatures of the files within the zipfile (*i.e.,* a "hash of hashes"), once again, just as in the '662 patent. (*Id.* at 9; Ex. 1004; *compare* '662 patent, col. 13, ll. 51-62 and Figure 10B; Ex. 1001.) FWKCS's contents-signatures accordingly could be used both to identify compound data items, like zipfiles, and to separately identify the segments (*i.e.*, individual files) contained within them. (Kantor at Preface 2; Ex. 1004.)

Kantor also specifically addressed the issue of deleting files and zipfiles identified by their contents-signatures, and included in his system features for helping BBS system operators delete unwanted files. (*Id.*; Ex. 1004.) For example, a "MULTIS" list (which listed duplicate files detected on the system) provided operators with an efficient mechanism for marking duplicate file stored in the system and for identifying and recording which of those files the system operator had deleted. (*Id.* at 189-200; Ex. 1004; *compare* '662 patent, col. 19, l. 59 to col. 20, l. 16 and Fig. 27(b); Ex. 1001.) Similarly, an "Exclude" feature permitted operators to mark and record deletions of files and zipfiles (identified by their contents-signatures) they wanted deleted from the system. If a user tried to subsequently upload a new copy of an "Excluded" file – identified by a "contents-

signature – the Exclude feature would automatically delete it again. (Kantor at 81; Ex. 1004.)

Although Kantor addressed the unique identification of files stored by BBS systems, he did not specifically address the underlying storage system because that was a concern for the BBS, not Kantor's system. However, the concept of replicated storage systems was very well known in the art. For example, the *Coda* system, among others, provided network-based file replication. (*See* M. Satyanarayanan et al., "Coda: A Highly Available File System for a Distributed Workstation Environment," IEEE Transactions on Computers, vol. 39, no. 4 (April 1990), pp. 447-459 ("Satyanarayanan II"); Ex. 1026.) It would have been straightforward and obvious to use replicated storage, as taught by Coda and others, to provide more reliable storage for BBS files. A person of ordinary skill in the art would have found it obvious to apply the teachings of Satyanarayanan II to Kantor (*e.g.*, to increase the reliability and response time of requests for files stored by BBS systems).

Woodhill: The Woodhill patent provides still another example of the use of context- and location-independent identifiers for the same purposes as the '662 patent. Woodhill created a distributed storage system that used "Binary Object

Identifiers" to identify and access files, and to manage file back-ups, among other functions. (Woodhill at Abstract; Ex. 1005.) As Woodhill explains, a "Binary Object Identifier 74 [of Fig. 3] . . . is a unique identifier for each binary object to be backed up." (Woodhill at col. 4, ll. 45-47; Ex. 1005.) The Binary Object Identifiers included three fields – a CRC value, a LRC value, and a hash value – each calculated from all of, and only, the contents of the binary object. (Woodhill at col. 7, l. 64 - col. 8, l. 33; Ex. 1005.) As Woodhill emphasized, "[t]he critical feature to be recognized in creating a Binary Object Identifier 74 is that the identifier should be based on the contents of the binary object so that the Binary Object Identifier 74 changes when the contents of the binary object changes." (Woodhill at col. 8, ll. 58-62; Ex. 1005). In addition to the original copy stored on a given local computer, Woodhill created two further backup (i.e., replica) copies of each binary object by (1) "stor[ing] a . . . copy of every binary object. . . somewhere on the local area network 16 other than on the local computer 20 on which it normally resides" and (2) "transmit[ing] every new or changed binary object to the remote backup file server 12." (Woodhill at col. 9, ll. 30-38; Ex. 1005.)

During a given backup cycle, Woodhill processed files deleted from a local

computer by creating a Backup Queue Record with a File Status of "DELETED." (*See* Woodhill at col. 5, l. 46 – col. 6, l. 64; Ex. 1005.) Woodhill processed these Backup Queue Records, associated with the Binary Object Identifiers for the binary objects of the deleted file, to propagate the local deletion throughout the system. (Woodhill at col. 6, l. 59 – col. 7, l. 8; Ex. 1005.) Woodhill additionally marked the time of this deletion by setting a "Delete Date" field to the time of such processing. (Woodhill at col. 6, ll. 59-64; Ex. 1005.)

Although Woodhill plainly described the basic elements of the '662 patent, including the record-keeping features, to the extent PersonalWeb claims that Woodhill did not disclose any particular details of the challenged claim, these features were well known in the art. For example, Ritchie described the same techniques as the '662 patent two decades earlier. Ritchie's UNIX system stored "i-lists" comprised of "i-node" entries for the files of the system. (Ritchie at 368; Ex. 1003.) Each i-node in this list stored a "use count" for a given file, the i-node being deleted when the "use count" dropped to zero. (Ritchie at 369; Ex. 1003.) These reference count techniques are the precise techniques described in the '662 patent, and the same techniques which the applicants identified as support for challenged claim 30. Thus to the extent Woodhill does not anticipate not only the

challenged claim but also the specific teachings of the '662 patent, the combination of Woodhill and Ritchie surely does. A person of ordinary skill in the art would have found it more than obvious to apply the teachings of Ritchie to Woodhill because Ritchie's technique is well-known, widely used technique to manage and delete shared data. Ritchie provided express motivation to make this combination, by disclosing that its i-list is a "quite simple and rapid algorithm for checking the consistency of a file system." (Ritchie at 369; Ex 1003.) Such a combination of Ritchie with Woodhill would have been the application of Ritchie's known techniques to the known device of Woodhill, ready for improvement, to yield the predictable result of improving the reliability of a file system. (*See* Ritchie at 369; Ex. 1003.)

These prior art references provide just a handful of many examples of the use of content-based identifiers to perform basic file management functions, including the deletion of files. Indeed, the application of hash-based identifiers to these functions was so obvious that at least one commentator not only described the applications as "easy" but also posted these ideas publicly "to impede anyone who might independently have had the idea from patenting it." (Williams, "An algorithm for matching text (possibly original)", posted to the "comp.compression"

newsgroup on January 27, 1992; Ex. 1035; *see also* R. Williams, "An Introduction to Digest Algorithms," Rocksoft (Nov. 1994), at 13 (further describing potential uses for file management purposes of identifiers based on the hash of the contents of a block of data); Ex. 1036.)

In short, other than perhaps coining a new phrase – i.e., True Name – for a very old concept, there is absolutely nothing new disclosed or claimed in the '662 patent concerning the use of location-independent, content-based data identifiers.

VI. SPECIFIC GROUNDS FOR PETITION

Pursuant to Rule 42.104(b)(4)-(5) and Practice Guide Fed. Register Vol. 77, No. 27, page 6873 Petitioners have submitted claim charts in connection with this Petition (attached as Exhibits 1027-1029), which have been submitted in the pending litigation between Petitioner and PersonalWeb Technologies LLC. Those charts set forth Petitioners' position with respect to those references and demonstrate that the challenged claims are anticipated and/or unpatentable in view of each of them. Petitioner also submits herewith the Declaration of Dr. Douglas Clark (Ex. 1009), a Professor of Computer Science at Princeton University. Dr. Clark confirms that the charts identify representative subject matter in each reference that teaches each and every limitation of the challenged claims. He

likewise confirms how each claim is anticipated or, at a minimum, rendered obvious by the prior art.

A. Grounds of Invalidity for Challenged Claim 30 based on Browne as a Primary Reference

Ground 1: Browne Anticipates Challenged Claim 30

Browne was not referenced or discussed by the examiner during prosecution of the '662 patent.[10] It is prior art under at least 35 U.S.C. § 102(a) and anticipates claim 30 of the '662 patent.

Browne describes the Bulk File Distribution ("BFD") package developed by researchers at the University of Tennessee and Bell Laboratories as part of an effort to make scientific software easily accessible over the Internet. (Browne at 1, 6; Ex. 1002.) The BFD package is based on the concept of a "virtual repository," which is a distributed network of physical software repositories, each residing on a different file server. (Browne at 1–2; Ex. 1002.) Files are mirrored on multiple servers "to increase availability (e.g., if one site is unreachable, the software may be retrieved from a different site) and to prevent bottlenecks." (Browne at 2; Ex. 1002.)

[10] Browne is cited on the face of the '662 patent as one of the over 400 references. Browne played no role in the prosecution of the '662 patent.

Like the '662 patent, Browne begins by discussing the shortcomings of context- or location-dependent file identifiers. At the time, a virtual repository could be implemented using a Uniform Resource Locator (URL) to identify each file. (Browne at 2; Ex. 1002.)[11] The authors identify several problems with the use of location-based identifiers, such as URLs, to access virtual software repositories. Among other things, URLs are inadequate for ensuring the consistency of a software repository. (Browne at 2; Ex. 1002.) Moreover, a URL can only identify a single location; if a virtual repository offers multiple copies of the same file, each copy must be given its own URL. (Browne at 2; Ex. 1002.)

In order to address these shortcomings, Browne adopts the same solution that would be later proposed in the '662 patent: associating a unique identifier with the ***contents*** of a file, rather than with the ***location*** of the file. Indeed, Browne even uses the same terminology as the patent, referring to its file names as "location independent." In the BFD package, the identifier is called a "Location

[11] A URL is a character string, such as "http://www.netlib.org/index.html," that can be used to specify a transfer protocol ("HTTP"), a location ("www.netlib.org"), and a file name ("index.html"). (*See, e.g.*, T. Berners-Lee et al., "Uniform Resource Locators (URL)," Internet RFC 1738 (Dec. 1994); Ex. 1018.)

Independent File Name," or LIFN. (Browne at 3; Ex. 1002; *compare* '662 patent, col. 3, ll. 55-58 ("the identity of a data item is ***independent*** of its name, origin, ***location*** . . . .") (emphasis added); Ex. 1001.)

In Browne's preferred approach, the LIFN is computed as the MD5 hash of the contents of a file. (Browne at 6; Ex. 1002.) The MD5 algorithm provides a substantially unique fingerprint, meaning that two files with identical content will always have the same MD5 fingerprint, even if they are located on different servers, and even if the server administrators give them different names.[12] "Once a LIFN has been assigned to a particular sequence of bytes, that binding may not be changed." (Browne at 3; Ex. 1002.)

A LIFN database maintains an association between the the LIFNs and the file servers. This is "a simple key/data database in which the unique keys are LIFNs." (Browne at 6; Ex. 1002.) To be clear, the LIFN database is an exact counterpart to the True File Registry (TFR) of the '662 patent, which receives a

---

[12] The general syntax for the LIFN is "lifn:netlib:<signature>", referencing the file access protocol ("lifn," similar to the "http" protocol identifier in a URL), the server handling the request ("netlib"), and the unique MD5 hash used to identify a file[12] ("<signature>"). (Browne at 4, 6; Ex. 1002.)

True Name identifier and provides a list of file servers that store a copy of that file. (‘662 patent, col. 35, ll. 47–49; Ex. 1001.)

Every time a file is added to or deleted from the system, the LIFN database needs to be updated to reflect the changes. The LIFN servers “accept updates from file servers containing new locations for LIFNs, as well as ***requests to delete*** old LIFN-to-location mappings.” (Browne at 5 (emphasis added); Ex. 1002.) It is especially important to delete entries in the LIFN database as soon as a file is deleted from a server, to prevent clients from requesting the non-existent file from that server and thus receiving an error. “If a file server moves or deletes a file, then it would post that information . . . .” (Browne at 5; Ex. 1002.) “When a client program requests [a file] from the file server, the file server either returns a file that is correct for the given LIFN, or it returns an error indicating that the file corresponding to that LIFN was not found.” (Browne at 6; Ex. 1002.)

Browne also addresses compound data items (*e.g.*, related files meant to be used together) in the same manner as the ‘662 patent. Browne refers to these compound items as “resources,” and specifically addresses the need to ensure consistency between them. (Browne at 2, 5–6; Ex. 1002.) To achieve that goal, Browne computes a LIFN for a resource as a whole, by applying the MD5 hash

function twice: first to each of the component files individually, and then to the list of the resulting hashes (*i.e.,* a "hash of hashes"). (Browne at 6; Ex. 1002.)

As set forth in detail in the attached claim chart (Exhibit 1027), and as confirmed by Dr. Clark (Clark Declaration at ¶¶ 17-38; Ex. 1009), Browne anticipates claim 30 of the '662 patent. Claim 30 recites:

> 30. A computer-implemented deletion method operable in a file system comprising (i) a plurality of servers; (ii) a list indicating, for each of a plurality of files in the file system, a corresponding status,
>
> wherein, for each of a plurality of data items in the file system, said data items each consisting of a corresponding sequence of one or more parts; and
>
> wherein each data item has a corresponding digital data item identifier, said digital data item identifier for the data item being based, at least in part, on the contents of the data item, wherein two identical data items in the file system have the same digital data item identifier; and
>
> wherein each part is replicated on multiple servers of said plurality of servers; and
>
> wherein said list includes digital data item identifiers for data items for which changes are to be made in the file system,
>
> the method comprising the steps of:

> (A) obtaining a particular digital data item identifier of a particular data item, said particular digital data item identifier of said particular data item being obtained in response to an attempt to delete said particular data item in said file system;
>
> (B) updating a record in said list to reflect deletion of said particular data item from the file system, said record including the particular digital data item identifier to the list.

('662 patent, col. 43, ll. 28-55; Ex. 1001.)

As Dr. Clark confirms, Browne discloses a "computer-implemented deletion method" that is operable in a virtual distributed repository (a "file system" comprising "a plurality of servers"). (Clark Decl., ¶ 23; Ex. 1009; Browne at 1; Ex. 1002.) Browne discloses the use of a LIFN database, which is "a simple key/data database," (a "list" of entries), and each entry maps a file's LIFN <signature> to the list of locations that hold a copy of that file. (Clark Decl., ¶ 24; Ex. 1009; Browne at 6; Ex. 1002.) Therefore, the LIFN database indicates a "status" of each file in the system, *e.g.*, where on the network the file is stored. (Clark Decl., ¶ 24; Ex. 1009; Browne at 4; Ex. 1002.)

Dr. Clark further confirms that Browne discloses a plurality of resources ("data items"), each of which consists of a corresponding sequence of one or more

files ("parts"). (Clark Decl., ¶ 26; Ex. 1009; Browne at 6; Ex. 1002.) Each data item has a corresponding MD5 signature ("digital data item identifier"). (Clark Decl., ¶ 28; Ex. 1009; Browne at 6; Ex. 1002.) The MD5 <signature> is based on the contents of the data item, since it is computed by applying an MD5 hash function to the contents of the data item. (Clark Decl., ¶ 28; Ex. 1009; Browne at 4, 6; Ex. 1002.) Based on the well-known properties of the MD5 hash function, two identical data items will always have the same digital data item identifier. (Clark Decl., ¶ 29; Ex. 1009.)

Dr. Clark also confirms that Browne discloses mirroring each file on multiple servers ("each part is replicated on multiple servers of said plurality of servers"). (Clark Decl., ¶¶ 30-31; Ex. 1009; Browne at 2, 4, Fig. 2; Ex. 1002.) The LIFN database (including the recited "list") includes the MD5 <signatures> for each of the data items in the system. Browne is clear that the database is "keyed" by a LIFN and includes those data items for which changes are to be made in the file system. (Clark Decl., ¶ 33; Ex. 1009; Browne at 6; Ex. 1002.) Such changes include, for example, the deletion of a data item from a file server. (Clark Decl., ¶ 33; Ex. 1009; Browne at 5; Ex. 1002.)

Regarding the two method steps in claim 30, Dr. Clark confirms that Browne discloses step (A). (Clark Decl., ¶¶ 34-35; Ex. 1009.) Specifically, when a file server deletes its copy of a file, the corresponding entry in the LIFN database also needs to be deleted, so that clients will no longer request the file from that particular server. (Clark Decl., ¶ 35; Ex. 1009; Browne at 5; Ex. 1002.) To that purpose, the file server sends a request to the LIFN server to update the LIFN database. (Clark Decl., ¶ 36; Ex. 1009; Browne at 5; Ex. 1002.) Upon receiving that request ("in response to an attempt to delete" the data item), the LIFN server obtains the MD5 <signature> of the file being deleted ("a particular digital data item identifier of a particular data item"). (Clark Decl., ¶ 36; Ex. 1009; Browne at 5–6.) In fact, the database uses the LIFN as a "key" when processing queries. (Clark Decl., ¶ 36; Ex. 1009; Browne at 6; Ex. 1002.)

Dr. Clark also confirms that Browne discloses step (B) of the claimed method,." (Clark Decl., ¶¶ 37-38; Ex. 1009.) Specifically, in response to a deletion request from the file server, the LIFN server "delete[s] old LIFN-to-location mappings," making clear that the database record corresponding to the LIFN key I s updated to reflect that deletion. (Clark Decl., ¶ 38; Ex. 1009; Browne at 5; Ex. 1002.) This involves "updating a record" in the LIFN database that

corresponds to the MD5 <signature> of the file being deleted, to reflect the deletion of that file from the server. (Clark Decl., ¶ 38; Ex. 1009; Browne at 5; Ex. 1002.)

The attached chart (Exhibit 1027) and the Declaration of Dr. Clark (Exhibit, 1009, ¶¶ 17-38) set forth Petitioner's position identifying where and how Browne anticipates the challenged claims.

B. Grounds of Invalidity for Challenged Claim 30 based on Kantor as a Primary Reference

Ground 2: Kantor Renders Challenged Claim 30 Obvious in View of Satyanarayanan II

Kantor was not cited to the USPTO and not considered during prosecution of the '662 patent. It is prior art under at least 35 U.S.C. § 102(b) and renders claim 30 of the '662 patent obvious.

Kantor is a published manual that describes a software program called the Frederick W. Kantor Contents-Signature System Version 1.22 ("FWKCS"). (Kantor at Title Page; Ex. 1004.) Like the '662 patent, Kantor addresses the shortcomings of context- or location-dependent file identifiers. (Kantor at Preface 1; Ex. 1004; *compare* '662 patent, col. 2, ll. 45-65; Ex. 1001) These include the

"problem of duplicate files on electronic bulletin board systems" or BBSs.[13] (Kantor at Preface 1; Ex. 1004.) BBS users would unwittingly or intentionally upload files to a bulletin board, which the bulletin board already had. (*Id.*; Ex. 1004) Consequently, bulletin board operators "were paying for hardware to provide the capacity for these spurious [duplicate] files, and spending many hours trying to find and delete them." (*Id.* ; Ex. 1004)

Kantor, like Browne, uses the same solution that would be later proposed in the '662 patent: associating a unique identifier with the ***contents*** of a file, rather than with the ***location*** of the file. Kantor calls these identifiers "contents-signatures," and uses them to identify a file based only on the contents of a file, and not its name, location, or other characteristics. (*Id.*; Ex. 1004; *compare* '662 patent, col. 2, ll. 45-65; Ex. 1001.) These signatures can be used for various purposes, including, for example, identifying duplicate content already stored on the BBS system. (Kantor at Preface 2-3; Ex. 1004.)

---

[13] Before the World Wide Web, computers "dialed into" a file server or network of servers where users could exchange files or other information by uploading or downloading files.

FWKCS computes the contents-signature based on a function of the data in a file. (*See id.* at 7-8; Ex. 1004.) Specifically, the contents-signature is constructed with "the 32-bit CRC [cyclic redundancy check][14] of the file contents and the uncompressed file-length." (*Id.*; Ex. 1004.) The CRC and the file length (*i.e.*, file size) are both a function of the data contained in the file, and two files with the same content necessarily have the same contents-signature. (*Id.*; Ex. 1004.) In fact, Kantor uses the same technique as in the '662 patent, creating a contents-signature with a hash and a length value. (*Id.* at 7-8; Ex. 1004; *compare* '662 patent, col. 13, ll. 51-62 and Figure 10A; Ex. 1001.) Each contents-signature is location-independent and independent of the file's pathname, location, or context.

Kantor also creates identifiers for compound data items in the same manner as the '662 patent. Kantor explains that BBS users often bundled files into "zipfile" format, a well-known format for organizing related files into a single compound file. (*See* Kantor at Preface 2; Ex. 1004.) FWKCS generates "zipfile contents-signatures" for these zipfiles by computing the contents-signatures for each of the individual files within the zipfile, then hashing these contents-

[14] As Dr. Clark confirms, a CRC is a well-known hash function that calculates a value as a function of the file's contents. (Clark Decl., ¶ 42; Ex. 1009.)

signatures using an "addition modulo 2^32" hash[15] to create the zipfile contents-signature for the zipfile as a whole (*i.e.*, a "hash of hashes"). (*Id.* at 9; Ex. 1004.)

FWKCS provides many operations for working with these zipfile and file contents-signatures. For example, FWKCS computes the zipfile and file contents-signatures for all of the zipfiles and individual files in the system, and stores them in a master contents-signature list, such as "CSLIST.SRT," which is just like the "True Name Registry" of the '662 patent. (*Id.* at 18; Ex. 1004; *compare* '662 patent, col. 33, ll. 52-54; Ex. 1001). Kantor also specifically addresses the issue of deleting unwanted files and zipfiles using their contents-signatures, including

---

[15] As Dr. Clark confirms, "addition modulo 2^32" is another well-known hash function that uses addition to calculate a value based on a file's contents. (Clark Decl., ¶ 43; Ex. 1009; *see also* G. D. Knott, Hashing functions, The Computer Journal 18 (1975), vol. 3, at 268 (describing "common elementary hashing functions" including addition functions); Ex. 1011.) By adding together the values of the contents identifiers for the individual files, Kantor ensures that "the resulting [zipfile contents identifier] does not depend on the names of the files, the dates of the files, [or] the order in which they appear in the zipfile . . . ." (Kantor at 9; Ex. 1004.)

mechanisms both for deleting selected files already stored on the BBS (Kantor at 189-90; *see also* Preface at 1-2; Ex. 1004), and deleting selected duplicate or otherwise unwanted files when a user tries to add them to the BBS (*id.* at 81; Ex. 1004).

To mark files for deletion that are already stored on the system, Kantor describes a "MULTIS" list. (*Id.* at 189-90; Ex. 1004.) The system operator can run an "m" command, which scans the master contents-signature list CSLIST.SRT for duplicate contents-signatures and places them into the "MULTIS" list. (*Id.*; Ex. 1004.) The MULTIS lists duplicate contents-signatures together so the operator can see at a glance which files contain identical content and which zipfiles contain identical groups of files, even if the files have different pathnames. (*Id.*; Ex. 1004.) The operator can mark unwanted copies for deletion with the "d" flag. (*Id.*; Ex. 1004.) At any time, the user can run a deletion command ("FWKC17D") to process the MULTIS list and delete every file marked with the "d" flag. (*Id.* at 200; *see also id.* at 190; Ex. 1004.) Consequently, the MULTIS list reflects deletion of the particular files and zipfiles that have been marked for deletion. (*See id.* at 200; Ex. 1004.) Additionally, the deletion command also updates the

"deletion.log" file with the contents-signatures and pathnames of the deleted files. (*Id.*; Ex. 1004.)

To delete unwanted files as they are being added to the system, Kantor describes the "Exclude" feature. (*Id.* at 81 and 117; Ex. 1004.) An operator can use the Exclude command to identify files in the system that should be deleted immediately and that also should be deleted if a user attempts to uploads a file with the same content as the file marked to be excluded from the system. (*Id.*; Ex. 1004.) (In the art, the exclude feature acts like a "black list," identifying entities that should not be allowed on the system; in Kantor, this black list identifies files precisely with their contents signatures.) When a file or zipfile is marked as an "excluded" file, its entry in the master contents-signature list CSLIST.SRT is marked with the "x" flag, reflecting deletion of the file from the system and deletion of the file if it is ever uploaded to the system again. (*Id.*; *see also id.* at 154; Ex. 1004.)

Although Kantor addressed the unique identification of files stored by BBS systems, he did not specifically address the underlying storage system because that was a concern for the BBS, not Kantor's system. However, the *Coda* system, among others, provided network-based file replication. (Satyanarayanan II, at 447-

459; Ex. 1026.) It would have been straightforward and more than obvious to use replicated storage, as taught by Coda and others, to provide more reliable storage for the BBS's files. (Clark Decl., ¶ 47; Ex. 1009; Satyanarayanan II at 447, 450; Ex. 1026.) A person of ordinary skill in the art would have found it obvious to apply the teachings of Satyanarayanan II to Kantor, for example, to increase the reliability of files stored by BBS systems. (Clark Decl., ¶ 47; Ex. 1009.)

As set forth in detail in the attached claim chart (Exhibit 1028), and as confirmed by Dr. Clark (Clark Declaration, ¶¶ 39-47; Ex. 1009), Kantor renders obvious claim 30 of the '662 patent. For claim 30, quoted above in Section V.A, Dr. Clark confirms that Kantor discloses at least two computer-implemented deletion methods in a BBS file system, which includes a plurality of servers, and which uses a contents-signatures list (e.g., CSLIST.SRT) indicating a "status" for each of a plurality of files in the BBS, such as if a file is marked as a duplicate or excluded file for deletion. (Clark Decl., ¶ 46; Ex. 1009 ; Kantor at 81 and 189-90, Ex. 1004.)

Dr. Clark further confirms that Kantor discloses a plurality of files and zipfiles ("data items"), each of which consists of a corresponding sequence of one or more parts. (Clark Decl., ¶ 43 ; Ex. 1009 ; Kantor at Preface 2, Ex. 1004.) Each

file and zipfile in the system has a signature ("digital data item identifier") which is obtained by a function (the CRC hash for a file with one part, or the addition modulo 2^32 hash for a zipfile with one or more parts) that is "based, at least in part, on the contents of the data item." (Clark Decl., ¶ 43 ; Ex. 1009 ; Kantor at Preface 2 and 10-11, Ex. 1004.) Both functions ensure that "any two identical data items will have the same digital data item identifier." (Clark Decl., ¶ 43; Ex. 1009; Kantor at Preface 2 and 10-11, Ex. 1004.)

Dr. Clark further confirms Kantor discloses a list of data item identifiers for which changes are to be made in the system, as well as the specific method steps of claim 30, with both the "MULTIS" list and "Exclude" feature. With respect to the "MULTIS" list, Dr. Clark confirms that the "MULTIS" list includes contents-signatures for files and zipfiles "for which changes," such as deletions, "are to be made in the file system. (Clark Decl., ¶ 45; Ex. 1009; Kantor at 189-90, Ex. 1004.) When a system operator runs the deletion command FWKC17D (an "attempt to delete"), the system obtains the contents-signatures (the "particular digital data item identifiers") for files and zipfiles in the MULTIS list that are marked to be deleted. (Clark Decl., ¶ 45; Ex. 1009; Kantor at 189-90, Ex. 1004.) The MULTIS list reflects deletion of the file. (Clark Decl., ¶ 45; Ex. 1009; Kantor at 189-90, Ex.

1004.) Moreover, FWKCS "updates a record" (the "deleted.log" file) to reflect deletion of the file. (Clark Decl., ¶ 45; Ex. 1009; Kantor at 200, Ex. 1004.) Dr. Clark further confirms that both MULTIS and deleted.log include the particular contents-signature of the deletion file. (Clark Decl., ¶ 45; Ex. 1009; Kantor at 189-90 and 200, Ex. 1004.)

With respect to the "Exclude" feature, Dr. Clark confirms that the CSLIST.SRT list includes contents-signatures for files and zipfiles "for which changes," such as deletions, "are to be made in the file system. (Clark Decl., ¶ 46; Ex. 1009; Kantor at 81, Ex. 1004.) When a system operator runs the Exclude command (an "attempt to delete"), the system obtains the contents-signatures (the "particular digital data item identifiers") for files and zipfiles in the CSLIST.SRT list that should be excluded. (Clark Decl., ¶ 46; Ex. 1009; Kantor at 81, Ex. 1004.) The CSLIST.SRT list reflects "exclusion" ("deletion") of the file by marking the file with an "x" flag. (Clark Decl., ¶ 46; Ex. 1009; Kantor at 81, Ex. 1004.) Dr. Clark further confirms that the CSLIST.SRT list includes the particular contents-signature (the "particular digital data item identifiers") in the deletion record. (Clark Decl., ¶ 46; Ex. 1009; Kantor at 81, Ex. 1004.)

Although Kantor does not explicitly disclose files that are replicated on multiple servers, Dr. Clark confirms that Kantor combined with Satyanarayanan II teaches that each file (with one part) or each part of a zipfile can be replicated on multiple servers within the modified file system. (Clark Decl., ¶ 47; Ex. 1009; Satyanarayanan II at 447; Ex. 1026.)

The attached claim chart (Exhibit 1028) and Declaration of Dr. Clark (Exhibit 1009, ¶¶ 39-47) set forth Petitioner's position identifying where and how Kantor renders obvious the challenged claims.

C. Grounds of Invalidity for Challenged Claim 30 based on Woodhill as a Primary Reference

Ground 3: Woodhill Renders Challenged Claim 30 Obvious in View of Ritchie

Woodhill was considered during prosecution of the '662 patent, but was considered in combination with different art and for claims substantially different to the claim now being challenged.[16] Woodhill is prior art under at least 35 U.S.C.

[16] The claims considered in light of Woodhill lacked, for example, the list indicating the status of files, now at the core of the challenged claim. As described above, this limitation was added to the claims of the '662 patent only in the final amendment before the '662 patent was allowed. Furthermore, there is no indication in the file history that the Patent Office specifically considered

§ 102(e) and anticipates claim 1 of the '662 patent.

Woodhill discloses a distributed storage management system, with mechanisms for backing up and later restoring the files stored by each computer of the system. (*Woodhill* at col. 2, ll. 39-49; Ex. 1005.) Figure 1 of Woodhill shows some of the basic elements of the system:

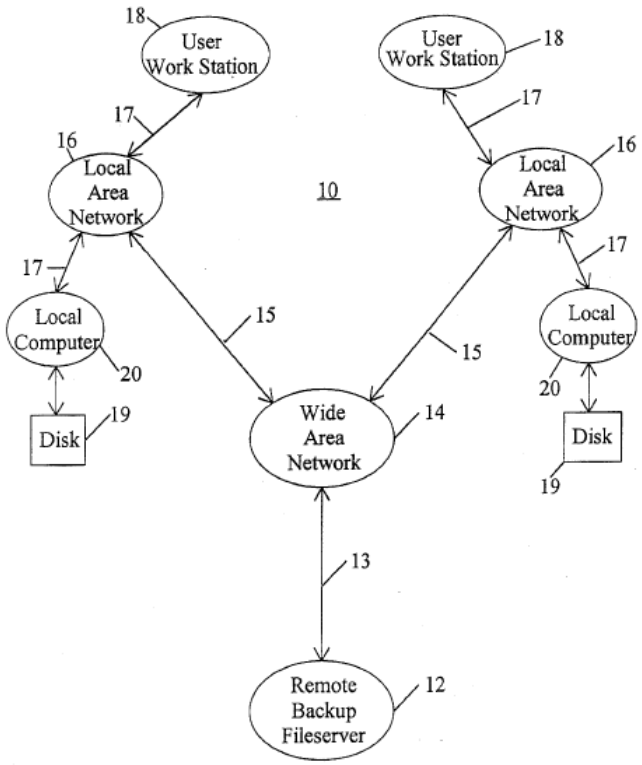

FIG. 1

To backup files or later access files, Woodhill breaks those files into "binary

---

Woodhill's Backup Queue Database or its associated 'File Status' and 'Delete Date' fields, now relevant to the challenged claim.

objects," having a size of one megabyte or less. (Woodhill at col. 4, ll. 12-30; Ex. 1005.) These files, and their component binary objects, are distributed across local computers 20 connected to local area networks 16. (Woodhill at col. 3, ll. 24-44; Ex. 1005.) These local computers 20 and their local area networks 16, in turn, are in communication with a remote backup file server 12 through a wide area network 14. (Woodhill at col. 3, ll. 12-27; Ex. 1005.) For each backup cycle, Woodhill scans the files of a local computer to identify new, changed, or deleted files since the last backup. (*See* Woodhill at col. 5, l. 46 – col. 6, l. 64; Ex 1005.) Backup copies (i.e., replicas) of files are then transmitted to both (1) another local computer 20 and (2) to the remote backup file server 12, so that they can be recovered if anything should go awry with the original copy. (Woodhill at col. 9, ll. 30-44; Ex 1005.) Only new or changed files are provided to these backup sites, and files deleted from the local computer 20 are eventually deleted from these backup sites. (Woodhill at col. 6, l. 56 – col. 7, l. 4 and col. 9, ll. 36-38; Ex. 1005.) Once files are backed up, the system allows a local computer 20 to recover them from a different local computer 20 or from the remote backup file server 12, as needed. (Woodhill at col. 9, ll. 39-44; Ex. 1005.)

For backup and restore operations, files are subdivided into one or more

binary objects. (Woodhill at col. 7, ll. 40-55; Ex. 1005.) To identify and compare binary objects as needed by these operations, Woodhill determines a Binary Object Identification Record, including a Binary Object Identifier, for each binary object. (Woodhill at col. 7, l. 60 – col. 8, l. 1; Ex. 1005.) Each "Binary Object Identifier is calculated from the contents of the data" of the corresponding binary object (*Id*. at col. 8, ll. 40-41; Ex. 1005). As Woodhill makes clear, the "***critical feature to be recognized in creating a Binary Object Identifier 74 is that the identifier should be based on the contents of the binary object*** so that the Binary Object Identifier 74 changes when the contents of the binary object changes." (Woodhill at col. 8, ll. 58-62 (emphasis added); Ex. 1005.) Accordingly, "the possibility of two different binary objects being assigned the same Binary Object Identifier 74 [is] very small," and each Binary Object Identifier 74 " uniquely identif[ies] a particular binary object." (Woodhill at col. 8, ll. 33-36; Ex. 1005.)

Figure 3 of Woodhill shows the Binary Object Identifier portion of a Binary Object Identification Record. (Woodhill at Figure 3; Ex. 1005.) This Binary Object Identifier is 128 bits long and includes Binary Object Size field, a Binary Object Cyclical Redundancy Check (CRC) field, a Binary Object Longitudinal Redundancy Check (LRC) field, and a Binary Object Hash field. (Woodhill at col.

7, l. 64 – col. 8, l. 4; Ex. 1005.) Each of the fields is "calculated from the contents of each binary object." (Woodhill at col. 8, ll. 1- 33; Ex. 1005). Binary Object Identifiers 74, like the True Names of the '662 patent, are thus based on the contents of the binary objects. (*Cf.* '662 patent, Figure 10A (calculating True Name based on MD5 hash of data item and length of data item); Ex. 1001).

To perform a backup cycle, each local computer 20 executes the Distributed Storage Manager program. (Woodhill at col. 5, ll. 3-20; Ex. 1005.) To begin the process, the program populates a Backup Queue Database with Backup Queue Records for processing during the present backup cycle. (Woodhill at col. 5, ll. 13-16; Ex. 1005.) The initial set of Backup Queue Records for each backup cycle is populated using the most recent Backup Instance Record for each file (i.e., from records created during the previous backup cycle). (Woodhill at col. 5, ll. 20-25; Ex. 1005.) Each Backup Queue Record corresponds to a file and stores a "File Status" field, having possible values of "NEW," "CHANGED," or "DELETED." (Woodhill at col. 4, ll. 48-61; Ex. 1005.) Figure 4 of Woodhill shows the structure of each Backup Queue Record:

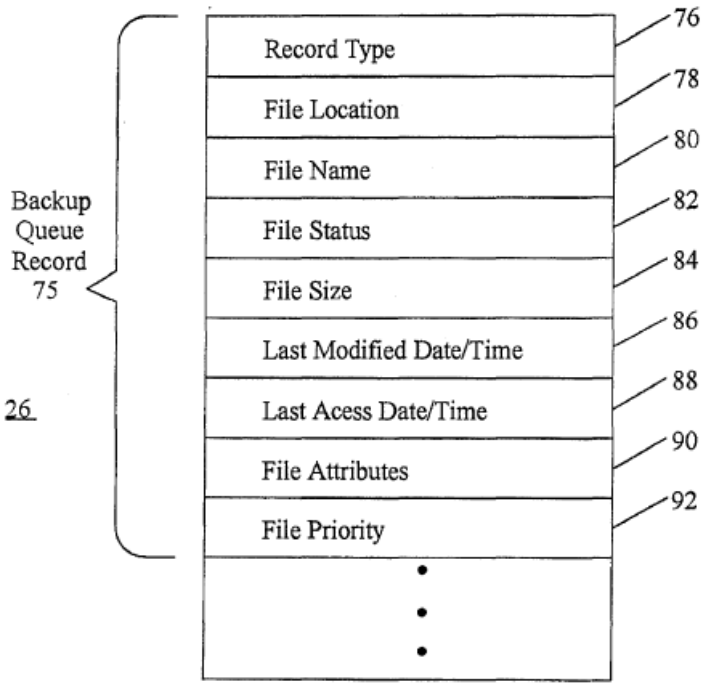

FIG. 4

As initially populated, the "File Status" field for every Backup Queue Record is set to "DELETED." (Woodhill at col. 5, ll. 30-32; Ex. 1005.) This status is then updated by "scan[ing] all disk drives 19 on the local computer 20 that are to be backed up." (Woodhill at col. 5, ll. 46-49; Ex. 1005.) For files found during the scan that have been modified since the previous backup, the "File Status" is changed to "MODIFIED." (Woodhill at col. 6, ll. 21-32; Ex. 1005.) If a file found during the scan is unchanged since the previous backup, the Backup Queue Record is deleted because the file does not need to be backed up during the

present cycle. (Woodhill at col. 6, ll. 35-39; Ex. 1005.) For files newly identified during this scan (e.g., files without a Backup Queue Record), a new Backup Queue Record is created with a "File Status" of "NEW." (Woodhill at col. 6, ll. 6-12; Ex. 1005.)

If a file for which a Backup Queue Record exists is not found during the scan (e.g., because the file was previously backed up, but has now been deleted), the "File Status" is left as "DELETED." (*See* Woodhill at col. 5, l. 30 – col. 6, l. 59; Ex. 1005.) In this case, to reflect deletion of the file, the "Delete Date field 56 in the most recent Backup Instance Record 42 associated with the file identified by the Backup Queue Record 75 currently being processed is set to the current date." (Woodhill at col. 6, ll. 59-64; Ex 1005.) Also, a " list of all Binary Object Identification Records 58 associated with the Backup Instance Record 42 for the file identified by the Backup Queue Record 75 currently being processed is placed in a delete queue (not shown) that will be used by Distributed Storage Manager program 24 to delete all Binary Object Identification Records 58 for binary objects that have been deleted from the disk drives 19 of local computer 20." (Woodhill at col. 6, l. 64 – col. 7, l. 4; Ex 1005.)

Although Woodhill plainly described the basic elements of the '662 patent,

including the record-keeping features, to the extent PersonalWeb claims that Woodhill did not disclose any particular details of the challenged claim, these features were well known in the art. For example, Ritchie described the UNIX system, which managed and tracked the status of files using the same techniques as the '662 patent. (*See* D. Ritchie et al., "The UNIX Time-Sharing system," Communications of the ACM, vol. 17, no. 7 (July 1974), pp. 365-375 ("Ritchie"); Ex. 1003.) Ritchie described a list, which he called an "i-list," that was managed by a UNIX system to store information describing each file of that system. (Ritchie at 368; Ex 1003.) Ritchie called each entry in this list an "i-node." (Ritchie at 368; Ex 1003.) "When a new file is created, an i-node is allocated for it and a directory entry is made which contains the name of the file and the i-node number." (Ritchie at 369; Ex 1003.) As UNIX allows users to create "a link to an existing file," but with a "new name," each i-node stored "[t]he number of links to the file, that is, the number of times it appears in a directory." (Ritchie at 369; Ex 1003.) The deletion of a file was reflected by "decrementing the link-count of the i-node specified by its directory entry," and, if the link-count dropped to zero, deallocating the i-node itself (Ritchie at 369; Ex 1003.) The solution described by Ritchie for the UNIX system tracks the approach of the '662 patent to a similar

problem. (*Cf.* '662 patent, col. 21, l. 65 – col. 22, l. 2 (describes decrementing the "use count" of a True File when its "use count is greater than one" and deleting the True File when its use count would otherwise be zero); Ex. 1001.)

Ritchie's UNIX system, like Woodhill, tracked the status of files stored by its system. As Dr. Clark confirms, the combination of Woodhill and Ritchie is more than obvious and would not only have anticipated the claims but also the particular techniques described in the '662 specification. (Clark Decl., ¶¶ 56-58; Ex. 1009.) It would have been obvious to combine the UNIX system's i-lists with Woodhill to enable efficient management and tracking of file status. (Clark Decl., ¶¶ 56-58; Ex. 1009.) Ritchie provides express motivation to make this combination, by disclosing that its i-list enables a "quite simple and rapid algorithm for checking the consistency of a file system." (Clark Decl., ¶ 56; Ex. 1009; Ritchie at 369; Ex 1003.) A person of ordinary skill in the art, exercising ordinary creativity, would have been motivated to combine the i-list taught by the UNIX system with Woodhill's backup system. (Clark Decl., ¶ 56; Ex. 1009.)

As set forth in detail in the attached claim chart (Exhibit 1029), and as confirmed by Dr. Clark (Clark Decl., ¶¶ 56-58; Ex. 1009.), Woodhill in view of Ritchie renders obvious claim 30 of the '662 patent (quoted above in Section V.A).

As Dr. Clark confirms, Woodhill discloses a network of local computers and a remote backup file server. (Clark Decl., ¶ 50; Ex. 1009; Woodhill at col. 3, ll. 24-44; Ex. 1005.) Woodhill further discloses a Backup Queue Database with a list of Backup Queue Records, each having an associated file, and each storing a "File Status" field for that file (a "list" indicating for each of the files in the file system a "corresponding status"). (Clark Decl., ¶¶ 53-54; Ex. 1009; Woodhill at col. 4, ll. 48-61; Ex. 1005.) Dr. Clark confirms that it would be obvious to further associate each file with an i-node, for example, stored in an i-list, as disclosed by Ritchie (another example of a 'list" indicating for each of the files in the file system a "corresponding status"). (Clark Decl., ¶¶ 56-58; Ex. 1009, Ritchie at 368; Ex 1003.) Each i-node stores a "link-count" status for its corresponding file. (Clark Decl., ¶ 57; Ex. 1009; Ritchie at 369; Ex 1003.)

Dr. Clark further confirms that Woodhill discloses a plurality of binary objects ("data items"). Each of Woodhill's Backup Queue Records corresponds to a file, and each file corresponds to one or more binary objects ("data items"). (Clark Decl., ¶ 53; Ex. 1009; Woodhill at col. 4, ll. 12-30, col. 5, ll. 3-20, and col. 4, ll. 48-61; Ex. 1005.) Each binary object consists of one or more granules (a "corresponding sequence of one or more parts"). (Clark Decl., ¶ 51; Ex. 1009;

Woodhill at col. 15, ll. 13-30; Ex. 1005.)

Dr. Clark confirms that each binary object has a corresponding Binary Object Identifier (a "digital data item identifier"), and that each identifier is calculated based a hash function of its corresponding binary object (based on "the contents of the data item"). (Clark Decl., ¶ 52; Ex. 1009; Woodhill at col. 7, l. 60 – col. 8, l. 1; Ex. 1005.) Dr. Clark further confirms that "two identical data items in the file system have the same digital data item identifier" because only the data of a binary object is used to calculated its Binary Object Identifier, and so two binary objects with the same data (i.e., identical data items) will necessarily have the same Binary Object Identifier. (Clark Decl., ¶ 52; Ex. 1009; Woodhill at col. 8, ll. 1-31; Ex. 1005.)

Dr. Clark also confirms that each local computer's binary objects (and thus their corresponding granule data) are stored on a second local computer and on the remote backup file server (each part is "replicated on multiple servers"). (Clark Decl., ¶¶ 53-54; Ex. 1009; Woodhill at col. 9, ll. 30-44; Ex. 1005.) Moreover, Dr. Clark confirms that each Backup Queue Record is associated with a list of all Binary Object Identification Records, including each binary object's Binary Object Identifier (a "digital data item identifier") for the file identified by the Backup

Queue Record). (Clark Decl., ¶ 55; Ex. 1009; Woodhill at col. 6, l. 64 – col. 7, l. 4; Ex. 1005.) The Backup Queue Records identify binary objects for which their Binary Object Identification Records are to be deleted ("for which changes are to be made in the file system"). (Clark Decl., ¶ 55; Ex. 1009; Woodhill at col. 4, ll. 48-61; Ex. 1005.)

Regarding the two method steps in claim 30, Dr. Clark confirms that Woodhill discloses step (A). Specifically, it discloses obtaining a Binary Object Identifier for a binary object ("a particular digital data item identifier of a particular data item") and that it does so in response to a user's attempt to delete a file from a local computer ("in response to an attempt to delete said particular data item in said file system"). The Backup Queue Record for a given file, associated with the Binary Object Identifiers for that file, is created during a backup cycle after a user has deleted a file from a local computer. (Clark Decl., ¶¶ 54-55; Ex. 1009; Woodhill at col. 6, l. 56 – col. 7, l. 4; Ex. 1005.)

Woodhill also discloses step (B). The Delete Date of the most recent Backup Instance Record associated with the Backup Queue Record being processed is updated to reflect deletion of the a data item ("updating a record in said list to reflect deletion of said particular data item"). (Clark Decl., ¶ 54; Ex.

1009; Woodhill at col. 6, ll. 59-64; Ex. 1005.) Additionally, Dr. Clark confirms that decrementing the "link-count" field of an associated i-node, and then deleting the i-node itself if the "link-count" has dropped to zero, would be a further "update[e] to a record in said list." (Clark Decl., ¶ 57; Ex. 1009; Ritchie at 369; Ex. 1003.)

The attached claim chart (Exhibit 1029) and the Declaration of Dr. Clark (Exhibit 1009, ¶¶ 48-58) set forth Petitioner's position identifying where and how Woodhill in view of Ritchie renders obvious the challenged claim.

VII. CONCLUSION

Based on the foregoing, it is clear that claim 30 of the '662 Patent recites subject matter that is either anticipated or obvious. The arguments cited above were never considered by the original Patent Examiner, and if it had been the '662 patent would not have issued. The Petitioner requests institution of an *inter partes* review to cancel this claim.

Respectfully Submitted,

/David L. Cavanaugh/
David L. Cavanaugh
Registration No. 36,476

CERTIFICATE OF SERVICE

I hereby certify that, on December 17, 2012, I caused a true and correct copy of the foregoing materials:

- Petition for *Inter Partes Review* of U.S. Patent No. 7,949,662
- Exhibits 1001-1036
- Fee Summary Page
- EMC Corp. Power of Attorney

to be served via Federal Express on the following attorney of record as listed on PAIR:

Davidson Berquist Jackson & Gowdey, LLP

Attn: Brian Siritzky, Ph.D.

4300 Wilson Blvd., 7th Floor

Arlington, Virginia 22203

/David L. Cavanaugh/

David L. Cavanaugh
Registration No. 36,476

Table of Exhibits for U. S. Patent 7,949,662 Petition for *Inter Partes* Review

Exhibit	Description
1001.	U.S. Patent No. 7,949,662
1002.	S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," University of Tennessee Technical Report CS-95-278 (Feb. 1995)
1003.	Ritchie and Thompson, The UNIX "Time-Sharing System," Comm. Of the ACM, vol. 17, No. 7 (July 1974)
1004.	Kantor, "The Frederick W. Kantor Contents-Signature System Version 1.22," FWKCS122.REF (August 10, 1993)
1005.	Woodhill et al., U.S. Patent No. 5,649,196, entitled "System and Method For Distributed Storage Management on Networked Computer Systems Using Binary Object Identifiers"
1006.	S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," http://www.netlib.org/utk/papers/lifn/main.html (Nov. 11, 1994)
1007.	K. Moore et al., "An Architecture for Bulk File Distribution," Network Working Group Internet Draft (July 27, 1994)
1008.	Chart of Patent Family Members
1009.	Declaration of Dr. Douglas Clark a Professor of Computer Science at Princeton University
1010.	Banisar et al., The Third CPSR Cryptography and Privacy Conference at 509 (1993)

1011.	G. D. Knott, Hashing functions, The Computer Journal 18 (1975), no. 3, p. 265.
1012.	R. Rivest, "The MD5 Message-Digest Algorithm," Internet RFC 1321 (Apr. 1992)
1013.	McGraw-Hill Dictionary of Scientific and Technical Terms, (4th ed., 1989)
1014.	B. Kaliski, "A Survey of Encryption Standards, " IEEE Micro (Dec. 1993)
1015.	Rabin, Fingerprinting by Random Polynomials, Center for Research in Computing Technology, Harvard University, Report TR-15-81
1016.	U. Manber, "Finding Similar Files in a Large File System", University of Arizonia Technical Report (1994)
1017.	D.R. McGregor and J.A. Mariani 'Fingerprinting' – A Technique for File Identification and Maintenance, 12 Software Practice & Experience 1165 (1982)
1018.	T. Berners-Lee et al., "Uniform Resource Locators (URL)," Internet RFC 1738 (Dec. 1994)
1019.	U. S. Patent 6,415, 280 Prosecution History, Response (August 22, 2001)
1020.	EP Pub. No. EP0826181A1 Prosecution History, Annex to the communication dated May 8, 2009
1021.	EP Pub. No. EP0826181A1 Prosecution History, Reply to communication from the Examining Division dated November 18, 2009
1022.	EP Pub. No. EP0826181A1 Prosecution History, Annex to the communication dated March 14, 2012

1023.	EP Pub. No. EP0826181A1 Prosecution History, Closing of Application dated June 14, 2012
1024.	U.S. Patent 7,949,662 Prosecution History, Amendment as filed on January 13, 2011
1025.	U.S. Patent 7,949,662 Prosecution History, Notice of Allowance of April 6, 2011
1026.	M. Satyanarayanan et al., "Coda: A Highly Available File System for a Distributed Workstation Environment, " IEEE Transactions on Computers, Vol. 39. No. 4 (April 1990) ("Satyanarayanan II")
1027.	Claim Chart, Invalidity Claim Chart in view of LIFN ("Browne")
1028.	Claim Chart, Invalidity Claim Chart in view of FWKCS Contents – signature System Version 1.22 ("Kantor")
1029.	Claim Chart, Invalidity Claim Chart in view of Woodhill
1030.	Merkle, U.S. Patent No 4,309,569, entitled "Method of Providing Digital Signatures," filed Sept. 5, 1979
1031.	Lampson and Sproull, "An Open Operating System for a Single-User Machine," ACM Operating Review (December 1979)
1032.	A. Tanenbaum, "Operating Systems: Design and Implementation", Prentice Hall, (1987)
1033.	E. Babb, "Implementing a Relational Database by Means of Specialized Hardware," ACM Transactions on Database Systems, Vol. 4, No.1, at 2-4, March 1979
1034.	D. Bitton and D. DeWitt, "Duplicate Record Elimination in Large Data Files," ACM Transactions on Database Systems, Vol. 8, No. 2, at 255 – 265 (June 1983)

1035.	R. Williams, "An algorithm for matching text (possibly original)," posted to the "comp.compression" newsgroup on January 27, 1992;
1036.	R. Williams, "An Introduction to Digest Algorithms," Rocksoft (Nov. 1994)